Pedagogická fakulta
Faculty of Education

Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky

Bakalářská práce

Vývoj aplikací pro Windows Phone „Mango“

Vypracoval: Zdeněk Valdauf, DiS.
Vedoucí práce: PaedDr. Petr Pexa, Ph.D
České Budějovice 2013

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
Fakulta pedagogická
Akademický rok: **2011/2012**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Zdeněk VALDAUF**
Osobní číslo:	**P10378**
Studijní program:	**B7507 Specializace v pedagogice**
Studijní obor:	**Informační technologie ve vzdělávání**
Název tématu:	**Vývoj aplikací pro Windows Phone "Mango"**
Zadávající katedra:	**Katedra informatiky**

Zásady pro vypracování:

V bakalářské práci bude zpracována problematika vývoje aplikací pro nový operační systém mobilních zařízení firmy Microsoft - Windows Phone 7.5 (Mango). Cílem bude představit a porovnat technologie vývoje mobilních aplikací v Silverlight a XNA frameworku, dále popsán a ukázán vývoj aplikací jak pro začínající programátory tak pokročilejší vývojáře. Součástí práce bude sada ukázkových aplikací pro Windows Phone a jedna plnohodnotná aplikace s dostupností na aplikačním portálu MarketPlace.

Rozsah grafických prací:	**CD ROM**
Rozsah pracovní zprávy:	**60**
Forma zpracování bakalářské práce:	**tištěná**
Seznam odborné literatury:	**viz příloha**

Vedoucí bakalářské práce:	**PaedDr. Petr Pexa, Ph.D.** Katedra informatiky
Datum zadání bakalářské práce:	**12. dubna 2012**
Termín odevzdání bakalářské práce:	**26. dubna 2013**

Mgr. Michal Vančura, Ph.D.
děkan

L.S.

doc. PaedDr. Jiří Vaníček, Ph.D.
vedoucí katedry

V Českých Budějovicích dne 12. dubna 2012

Příloha zadání bakalářské práce

Seznam odborné literatury:

1. App Hub: Windows Phone development quickstarts. App Hub [online]. 2012 [cit. 2012-03-24]. Dostupné z: http://create.msdn.com/en-us/education/quickstarts
2. MSDN: Windows Phone - začínáme. MSDN [online]. 2012 [cit. 2012-03-24] Dostupné z: http://blogs.msdn.com/b/vyvojari/archive/2011/08/16/windows-phone-zaciname.aspx
3. Vbnet: XNA. Vbnet [online]. 2008 [cit. 2012-03-24]. Dostupné z: http://www.vbnet.cz/kategorie–16.aspx
4. Vbnet: Blog - Slavicek. Vbnet [online]. 2012 [cit. 2012-03-24]. Dostupné z: http://vbnet.cz/blog–slavicek.aspx
5. Smartmania: Seriál Tomáše Slavíčka. Smartmania [online]. 2012 [cit. 2012-03-24]. Dostupné z: http://smartmania.cz/tag/programovani-152
6. Silverlight: Budování Windows Phone 7 aplikace. Silverlight [online]. 2012 [cit. 2012-03-24]. Dostupné z: http://www.silverlight.net/archives/tutorials/building-a-windows-phone-7-application
7. Smartmania: Vyvíjíme pro Windows Phone. Smartmania [online]. 01-01-2005 [cit. 2012-03-24]. Dostupné z: http://smartmania.cz/tag/xna-190
8. MSDN Magazine: Your First Windows Phone Application. MSDN [online] 2012 [cit. 2012-03-24]. Dostupné z: http://msdn.microsoft.com/en-us/magazine/hh708749.aspx

Prohlášení

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně, pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě fakultou elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích dne

Poděkování

Rád bych poděkoval PaedDr. Petru Pexovi za odborné vedení, připomínky a cenné rady při zpracování této práce.

Anotace

V práci se zabývám vývojem mobilních aplikací pro téměř nový operační systém Windows Phone 7.5. V práci je popsaná historie tohoto operačního systému. Systém bude představen vývojářům, kteří chtějí zjistit specifika a možnosti vývoje pro tento mobilní operační systém. Budou zde také popsány komponenty a technologie pro vývoj. Dále v práci nalezneme podrobnou kapitolu zaměřenou na samostatný vývoj, kde je popsáno vše od stažení potřebného programového vybavení, až po publikování aplikace na MarketPlace. Součástí práce bude jedna plnohodnotná aplikace s dostupností na aplikačním portálu MarketPlace, která bude v práci podrobněji rozebrána.

Klíčová slova: Mobilní aplikace, Silverlight, XNA, XAML, Windows Phone, Visual Studio C#, Microsoft, Marketplace.

Abstract

The work deals with the development of mobile applications for nearly new operating system Windows Phone 7.5 The work described the history of the operating system. The system will be introduced for developers who want to find out specifics and development opportunities for this operating system. There will also be described components and technology for development. Further work will find a detailed chapter focused on individual development, which describes everything from how to download the necessary software to the publication of the application on MarketPlace. Part of the work will be one full-featured application with access to the Application Portal MarketPlace, which will be discussed in more detail in the work.

Keywords: Mobile application, Silverlight, XNA, XAML, Windows Phone, Visual Studio C#, Microsoft, Marketplace.

Obsah

1 Úvod

Cílem mé bakalářské práce je seznámit čtenáře s vývojem aplikací pro mobilní operační systém Windows Phone od společnosti Microsoft. Tato práce pomůže především programátorům, kteří chtějí zjistit specifika a úskalí vývoje pro tento operační systém.

Tuto práci jsem si vybral kvůli své zálibě v mobilních telefonech. Telefon s operačním systémem Windows Phone vlastním už od příchodu první verze, tudíž jsem již měl co dočinění s experimentováním s tímto systémem. Tento systém mě tak zaujal, že jsem se musel do problematiky vývoje na tento systém ponořit hlouběji.

V této bakalářské práci se seznámíme s historií mobilního operačního systému Windows. Dále zde popisuji vývoj aplikací pro mobilní operační systém Windows Phone od stažení potřebného programového vybavení, až po publikaci aplikace na publikační portál MarketPlace[1].

V praktické části práce jsem si vybral naprogramování a popsání aplikace pro Katedru informatiky Pedagogické fakulty Jihočeské univerzity, která bude sloužit jako rozcestník pro uchazeče o studium, stávající studenty či pracovníky katedry. V této kapitole ukazuji všechny důležité metody pro chod aplikace.

[1] PC verze MarketPlace:http://www.windowsphone.com/en-us/store?signin=true

2 Historie mobilního operačního systému Windows

2.1 Windows Mobile

Windows Mobile je jedním z nejstarších operačních systémů od firmy Microsoft, který je založen na Windows CE. Tento operační systém je určen pro mobilní zařízení. Za deset let vývoje uvedla firma Microsoft již přes dvanáct verzí systému. Všechny verze si zakládají na obrazovce tzv. „Domů“ která zobrazuje všechny důležité informace – události, čas, kalendář, stav signálu, e-mailové zprávy, úkoly atd. Již první verze systému obsahovala Office Mobile známé jako Microsoft Office (Word, Excel, Outlook), dále obsahovala známé aplikace typu Internet Explorer, Outlook, Windows Media Player (podporované formáty AVI, MP3, WMV, WMA).

Dne 19. dubna 2000 došlo k představení operačního systému s názvem Pocket PC 2000. Svým vzhledem se podobal klasickým desktopovým verzím Windows 98 a Windows 2000. Byl určený především pro kapesní počítače, až na pár výjimek. Tento systém byl pro vývojáře velmi otevřený, ti měli značné možnosti pro jeho úpravy. Toto jim však zároveň komplikovalo situaci, neboť systém byl nestabilní, protože systém bylo možné spustit skoro na jakémkoliv hardwarovém základu. Druhá verze vyšla v říjnu 2001 pod názvem Pocket PC 2002. Tento systém se začal podobat Windows XP. S touto verzí se objevili i první smartphony[2].

Dne 23. června 2003 vyšla třetí verze označovaná jako Windows Mobile 2003. Specialitou bylo uvedení vůbec první hry pro mobilní operační systém – Jaebreaker.

[2]V českém překladu - chytrý telefon, který využívá pokročilý operační systém a aplikační rozhraní.

Jedna z revolučních verzí byla představena na konferenci v Las Vegas 9. května 2005. Jako první totiž používala NET Compaq Framework 1.0 SP2[3]. Tato verze také přinesla větší výdrž baterie díky střídání Flash a Ram paměti. Přibyla zde i nová aplikace do Office Mobile a to PowerPoint Mobile (prohlížeč formátu PPT). Dále přibyla podpora technologií Direct3D, DirectDraw a DirectShow[4] pro vývojáře.

Dne 12. února 2007 byla představena další významná verze Windows Mobile 6. Tato verze byla silně vázána na Windows Live a Exchange 2007[5]. Podporovala rozlišení 800x480. Vzhledově se podobá systému Windows Vista. V tomto systému přibyla podpora VoIP[6] a Windows Live. Přibyla zde i schopnost automatických aktualizací systému. Další menší aktualizace 6.1 byla představena 1. dubna 2008. Ta byla určena především pro dotekové telefony. Obsahovala pouze nějaké drobnosti např. interaktivního průvodce. A poslední verze systému Windows Mobile nesla označení 6.5. Microsoft si konečně všiml, že mu ujíždí vlak, tak začal pracovat na úplně novém systému. Jelikož vyvíjení nového systému trvalo příliš dlouho, vydal Microsoft narychlo ještě jednu „meziverzi“. Ta přinesla velmi pěkné prostředí a nový Internet Explorer 6. Pak ještě o něco později vyšlyverze 6.5.1 a 6.5.3, které s sebou nesly pár drobných vylepšení a Office Mobile 2010. A jako poslední vyšla verze 6.5.5, která přinášela opravdu nepatrné změny[1]

2.2 Windows Phone

Windows Phone je mobilní operační systém, který v momentální době nalezneme na telefonech. Tento systém byl vyvíjen již v roce 2004 a nesl označení „Photon“. Vývoj tohoto systému se velice táhl a nakonec byl zrušen.

---

[3] Je verzí .NET Frameworku, která je určena pro běh na mobilních zařízeních.

[4]Multimediální framework

[5] Účet na serveru Microsoft

[6]Technologie umožňující přenos digitalizovaného hlasu

O čtyři roky později se skupina vývojářů opět pustila do vývoje tohoto operačního systému.

V únoru 2010 vydal Microsoft tiskovou zprávu, která potvrzovala změnu z Windows Mobile na Windows Phone. Rovněž uvedla i společnosti, které budou vyrábět telefony s tímto operačním systémem (HTC, Dell, Samsung, LG). Systém byl představen 15. února 2010 na mobilním kongresu v Barceloně.

11. října 2010 představil SteveBallmer deset zařízení s operačním systémem Windows Phone 7. Uvedl zde i datum (21. října v Evropě a Austrálii, 8. listopadu v USA) začátku prodeje telefonů s tímto operačním systémem.

Hlavní novinkou je změna hlavní obrazovky. Zcela se změnilo klasické zobrazení Windows, jak je známe ze systému Windows XP. Nahradily je velké dlaždice nazývané „huby“. Každý tento hub představuje multimediální centrum, ve kterém nalezneme další funkce či aplikace. Ale i tento operační systém má několik špatných vlastností, např. zde zcela chybí možnost mít v telefonu paměťovou kartu, nebo veškerý obsah z telefonu nebo do telefonu lze kopírovat pouze pomocí programu Microsoft Zune[7].[1]Od svého prvního vydání mobilního operačního systému Windows Phone vyšlo již několik aktualizací. První aktualizace na tento operační systém, která byla pojmenovaná jako Nodo, nesla s sebou rychlejší systém a podporu kopírování a vkládání textu. Další aktualizace, která se na naše zařízení dostala v říjnu roku 2011, nesla název Mango. Po této aktualizaci se Windows Phone 7 začal označovat jako Windows Phone 7,5. Systém s touto aktualizací dostál významných změn, a to hlavně pro Českou Republiku, protože se objevila podpora češtiny a dále se objevilo více jak 500 nových funkcí. Jako poslední aktualizace, která stojí za zmínku, byla aktualizace na Windows Phone 7,8. Tato aktualizace přinesla hlavně přepracovanou startovací plochu. Tato

[7] Zune je software pro správu zařízení.

aktualizace měla spíše převést starší generaci systému do „novějšího kabátu“, bohužel jen po vzhledové stránce.

V říjnu roku 2012 vyšla nová generace Windows Phone označená číslicí 8. Tato verze přináší vcelku radikální změny a to především z pohledu programátorského. Byla totiž odstraněna XNA technologie jako hlavní freamwork pro vývoj. Je zde sice malá zpětná kompatibilita mezi oběma generacemi technologií, ale aplikace napsané ve Windows Phone 7 je nutné převést na Windows Phone 8.[2]

3 Minimální hardware požadavky na mobilní zařízení.

Každý telefon s tímto operačním systémem musí splňovat kritéria na hardware danáMicrosoftem. Minimální požadavky na systém jsou: Telefon musí mít minimální rozlišení displeje 480x800 pixelů, musí také podporovat rozhraní DirectX9[8] a musí podporovat čtyřbodový Multitouch.

Obrázek 1 - Harware požadavky[9]

Dále musí vlastnit podporu lokalizace telefonu – GPS[10], akcelerometr[11] či FM rádio.Musí vlastnit integrovaný fotoaparát, který musí mít 1mpx či více. Paměť musí být o minimální velikosti 256MB a Flash paměť[12] 8GB nebo více. Procesor musí být dvoujádrový a podporované čipy jsou jen MSM7X30, MSM8X55, QSD8X50 nebo vyšší. Jako poslední musí být telefon opatřen třemi hardware tlačítky – zpět – start – vyhledávání.[3]

[8] DirectX9: Je rozhraní potřebné pro hry a multimediální aplikace.

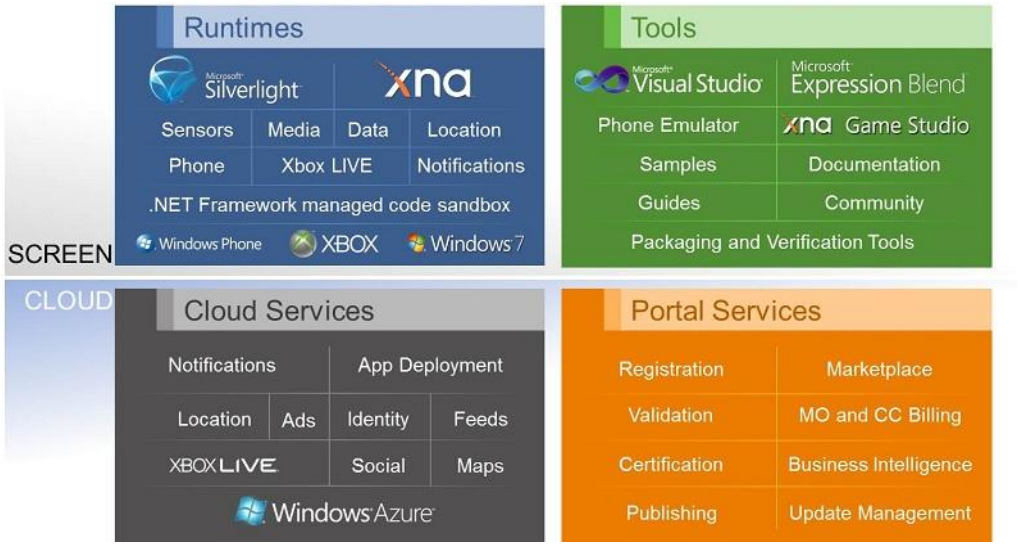
[9] Zdroj:http://wmpoweruser.com/microsoft-relaxes-wp7-hardware-requirements-enable-camera-less-phones/

[10]Globální polohový systém

[11]pohybový senzor

[12]elektricky programovatelná paměť s libovolným přístupem.

4 Architektura Windows Phone



Obrázek 2 - Architektura Windows Phone[13]

Architektura mobilního operačního systému Windows Phone se skládá ze čtyř hlavních komponent a to z:

- **Runtimes -** ve kterém jsou zahrnuty technologie Silvelight a XNA, dále je zde obsaženo ovládání senzoru lokací a umožňuje tak vytvářet složité aplikace a hry,
- **Tools** – obsahuje nástroje jako je Visual Studio a ExpresionBlend a všechny nástroje s nimi nainstalované,
- **CloudService** – slouží k využívání cloudových služeb jako je Xbox Live či Windows Azure,
- **PortalServices -** obsahujeMarketPlace sloužící k distribuci aplikací.

[13] Zdroj http://themobilityexpert.blogspot.cz/2012/06/wp7-platform-insights-part-1-overview.html

5 Technologie pro vývoj Windows Phone aplikací.

5.1 XNA

Microsoft XNA Framework je sada programovacích nástrojů určených pro vývoj 2D a 3D her .NET Framework je složen z knihoven a programového API, které slouží ke snížení množství kódu potřebného pro běžné herní úkoly. Záměrem rozhraní XNA Frameworku je umožnit programátorům zaměřit se více na obsah her, aniž by museli trávit velké množství času u herního engine a grafického rámce.

5.1.1 XNA Game Studio

Framework XNA může být integrován do Visual Studia, nebo ho Microsoft zpřístupňuje prostřednictvím softwarového balíku známého jako Studio Game XNA. Game Studio je programovacím prostředí, ve kterém se dají programovat a testovat rozpracované hry, aniž by bylo nutné otevřít samostatný kompilátor. Zatím bylo vydáno celkem pět vydání Studia Game XNA. V roce 2012 vyšel poslední XNA Game Studio 4.0 Refresh, který zahrnuje opravy chyb a podporu pro Windows Mobile 7.5 a Visual Basic.

5.1.2 Podporované platformy

Hry vytvořené pomocí Microsoft XNA Frameworku podporují Windows PC, Xbox 360, Windows Phone 7 a Windows Mobile 7.5. Plná podpora pro Windows Phone 8 není k dispozici v rozhraní XNA Framework. Od roku 2012bude možné spustit hry vyvinuté pomocí Windows Phone 7 nebo 7,5 jako starší aplikace na novější platformě.

5.1.3 XDK rozšíření

XDK rozšíření je známé jako XNA Studio Professional, je to volitelná součást pro vývojáře, kteří mají platnou licenci Xbox rozvoje a Microsoft Xbox

360 DevelopmentKit. Umožňuje vývojářům zahrnout i další herní prvky, mezi které patří např. ukládání úspěchů a online žebříčků, které jsou vyhrazeny pouze pro použití v licencovaných hrách Xbox.

5.1.4 XNA a Windows 8

Hry doposud vytvořené na XNA Frameworku jsou kompatibilní s Windows 8, ale od roku 2012 Framework XNA nelze použít k vytvoření herní aplikace pro Windows moderní rozhraní UI. Desktop hry vyvinuté s rozhraním XNA Framework mohou být uvedeny v Microsoft Store, i když přesměrují návštěvníky na developera internetových stránek pro nákup a stažení.

5.1.5 MonoGame

MonoGame je Open-source implementace v XNA Frameworku 4.0. Může dovolit vývojářům překompilovat XNA projekty pro publikaci jako moderní UI. Zároveň také umožňuje zveřejnění her v rozhraním XNA na jiných nepodporovaných platformách jako je Mac OS X, Apple iOS, Android, Linux a PlayStationSuite. MonoGame vyžaduje Visual Studio.[4]

5.2 Silverlight

Technologie Silverlight je také využívaná jako platforma pro vývoj aplikací na operační systém Windows Phone.Vhodná je pro psaní různých utilit nebo aplikací často napojených na internet. Vývojáři na této technologii oceňujípředevším její rychlost a stabilitu aplikací. Silverlight pro Windows Phone podporuje technologie známé z velkého Silverlightu, kterými jsou:

- Video a audio ve vysoké kvalitě za použití širokého spektra kodeků, DRM[14] a IIS Smoothstreamingu[15]

[14] Správa digitálních práv

[15]Live SmoothStreaming umožňuje přizpůsobivé streamování živých událostí klientům kompatibilních s Microsoft Silverlight.

- Funkce Deep Zoom[16] pro lepší zážitek z prohlížení textů a fotografií
- Animace vektorové i bitmapové grafiky

Silverlight může také přistupovat k jedinečným vlastnostem telefonu:

- Nativní funkce telefonu
- Využívání informací o pozici telefonu
- Akcelerometr pro snímání pohybu
- Dotykový display s podporou multi-touch[17]
- Hardwarové akcelerace videa a grafiky
- Kamera a mikrofon
- Oznámení typu PUSH[18]

Technologie Silverlight dokáže užívat XNA Framework pro přehrávání a nahrávání zvuku, dále dokáže přistupovat k specifickým knihovnám a dokonce i k Xbox LIVE[19]. Všechny tyto vlastnosti jsou snadno dosažitelné pouhým přidáním knihovny do vaší aplikace založené na technologii Silverlight. Po přidání knihovny do aplikace je už velice snadné tyto funkce využít. Technologie Silvelight pro Windows Phone podporuje funkci vyzkoušej/kup API, která umožňuje uživatelům ještě před koupí plné verze aplikace vyzkoušet tzv. trial verzi a až poté si zakoupit plnohodnotnou verzi aplikace.[5]

[16] Technologie pro efektivní přenos a prohlížení snímků.

[17] Vícedotykové ovládání

[18] Jsou nahrazením aplikací běžících na pozadí

[19] Účet od firmy Microsoft

6 Programové vybavení pro vývoj WP aplikací

V první řade pro vývoj aplikací na Windows Phone budeme potřebovat nástroje, které nalezneme v tzv. SDK (Software DevelopmentKid). Balík SDK obsahuje veškeré potřebné nástroje pro vývoj pro snadné použití. Popřípadě, pokud máte nainstalované Visual Studio 2010, ho jen doplní o nástroje pro vývoj aplikací na Windows Phone.

Windows Phone SDK je zcela zdarma ke stažení na adrese:https://dev.windows phone.com/en-us/downloadsdk

Nalezneme zde Windows Phone SDK 7.1. Jedná seo kompletní vývojový balík sloužící pro vývoj aplikací pro Windows Phone i pro vývoj her v rámci Xbox Live. Obsahuje vše, co je potřeba k plnohodnotnému vývoji aplikací.

Ve Windows Phone SDK balíku nalezneme:

- **Microsoft Visual Studio 2010 Express for Windows Phone**
  Vývojové prostředí Visual Studio 2010.
- **Windows PhoneEmulator**
  Emulátor pro telefon, který nahrazuje fyzické zařízení při testování.
- **Windows Phone SDK 7.1 Assemblies**
  Systémové a vývojové knihovny.
- **Silverlight 4 SDK and DRT**
  SDK pro Silverlight.
- **Windows Phone SDK 7.1 Extensionsfor XNA Game Studio 4.0**
  Podpora pro vývoj WP aplikací v XNA studiu
- **Microsoft ExpressionBlend SDK for Windows Phone OS 7.1**
  Nástroj pro vývoj uživatelského rozhraní.
- **WCF Data ServicesClientforWindowPhone**
  Podpora pro datové webové služby.
- **Microsoft Advertising SDK for Windows Phone**
  Podpora pro integraci reklam do aplikací.[6]

6.1 Systémové požadavky na počítač

Podporované operační systémy: Windows 7, Windows Vista
Windows ® Vista ® (x86 a x64) s ServicePack 2 - všechny verze kromě StarterEdition, Windows 7 (x86 a x64) - všechny verze kromě StarterEdition

Instalace vyžaduje 4 GB volného místa na disku na systémové jednotce a 3GB RAM.

Windows Phone Emulátor vyžaduje DirectX 10 nebo vyšší a Grafickou kartu s WDDM 1.1 driver.

Windows Phone SDK 7.1 je kompatibilní s poslední verzí Visual Studio 2010 SP1.[7]

Po instalaci Windows Phone SDK je váš počítač plně připraven pro vývoj aplikací na mobilní zařízení. Pro testování svých vytvořených aplikací využíváte emulátor, který je součásti SDK. Pro testování na fyzických zařízeních a šíření svých aplikacích je potřeba mít vytvořený vývojářský účet.

6.2 Vývojářský účet

Abychom mohli testovat aplikace a hry na fyzických zařízeních, nebo zveřejňovat aplikace na MarketPlace, musíme vlastnit vývojářský účet. Získání vývojářského účtu je jednoduché, vyžaduje registraci na APP HUB[20]. Registrace vyžaduje WindowsLive ID[21], které si zaregistrujete zde: http://www.live.com. V registraci je potřeba projít registračním formulářem, ve kterém máte na výběr registraci jako právnická osoba, fyzická osoba nebo jako student. Členský poplatek činí 99$ za rok. Pokud jste se zaregistrovali jako

[20] Zdroj: http://create.msdn.com

[21] Windows Live ID je e-mailová adresa a heslo sloužící k přihlášení následujících služeb Xbox Live, Zune, Skydrive, MSN…

student, máte vývojářský účet zdarma, jelikož máte možnost využít programDreamSpark[22],který ověřuje platnost karty ISIC[23].

Vývojářský účet nám umožňuje:

- vytvářet aplikace zdarma i placené,
- publikovat na MarketPlace neomezený počet placených aplikací,
- publikovat na MarketPlace až 100 bezplatných aplikací,
- odemknout až tři telefony pro vývoj a testování,
- získat přístup k vývojovým materiálůma přístup k vývojářské komunitě a podpoře vývoje.

[22] Program na http://www.dreamspark.com/

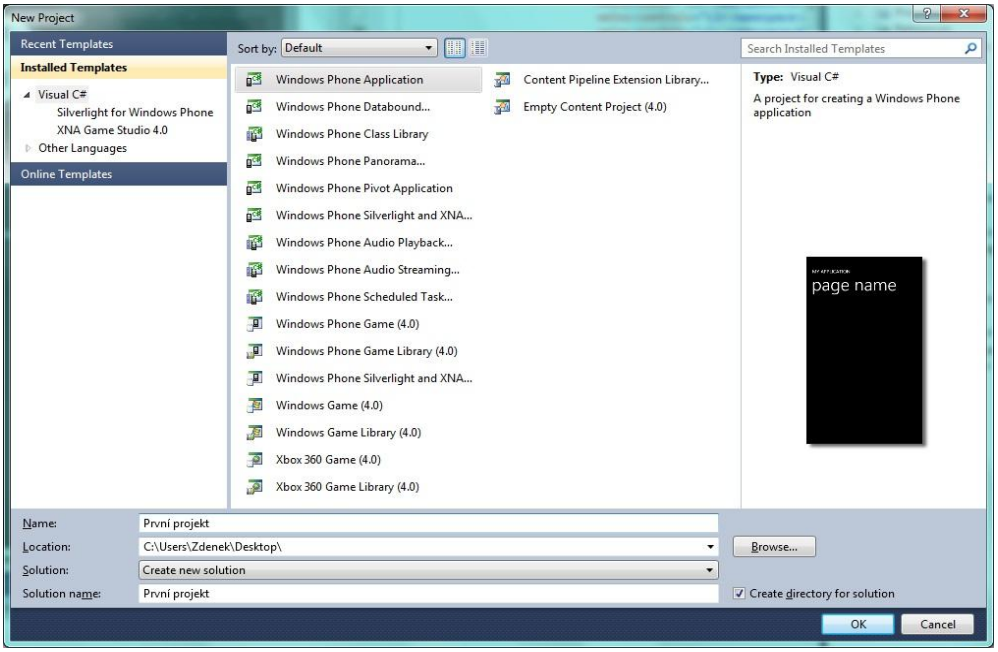
[23] Studentská karta

7 Vývoj aplikací

7.1 Vytvoření projektu

Zapneme program Visual Studio a v dialogové nabídce zvolíme záložku New Project. Zde si zvolíme architekturu vývojového prostředí. K dispozici máme dvě možnosti a to buď Silverlight nebo XNA freamwork. Každá možnost je určená pro jiné účely. Silverlight je vhodnější na klasické aplikace postavené na uživatelském rozhraní pro prezentaci dat. XNA freamwork je naopak lépe využitelný pro aplikace s vysokým podílem 2D/3D grafiky a pro herní prvky.

Pokud víme, která z nabízených technologií bude pro náš projekt vhodnější, tak si ji vybereme. Já jsem si vybral technologii Silverlight. Zde opět máme na výběr z několika konceptů aplikace, které nám slouží jako šablona a usnadňují nám práci. K dispozici máme tyto následující typy projektů:



Obrázek 3- Visual Studio: výběr projektu

Windows PhoneAplication

- Jedná se o základní šablonu, která obsahuje připravenou strukturu projektu s jednou stránkou uživatelského rozhraní.

Windows PhoneDataboundApplication

- Rozšířena šablona sloužící k napojení dat a zobrazení v seznamu.

Windows PhoneClassLibrary

- Prázdný projekt, který je připravený pro tvorbu tříd tvořících knihovnu Windows Phone.

Windows Phone Panorama Application

- Šablona, která vychází z vedle sebe umístěných panelů.

Windows PhoneSilverlight and XNA Application

- Šablona umožňující v jedné aplikaci integrovat jak prvky Silverlightu, tak prvky XNA freamoworku.

Windows Phone Audio Playback Agent

- Knihovna sloužící pro přehrávání audio souborů.

Windows Phone Audio Streaming Agent

- Knihovna, jež se specializuje na přehrávání streamovaného videa.

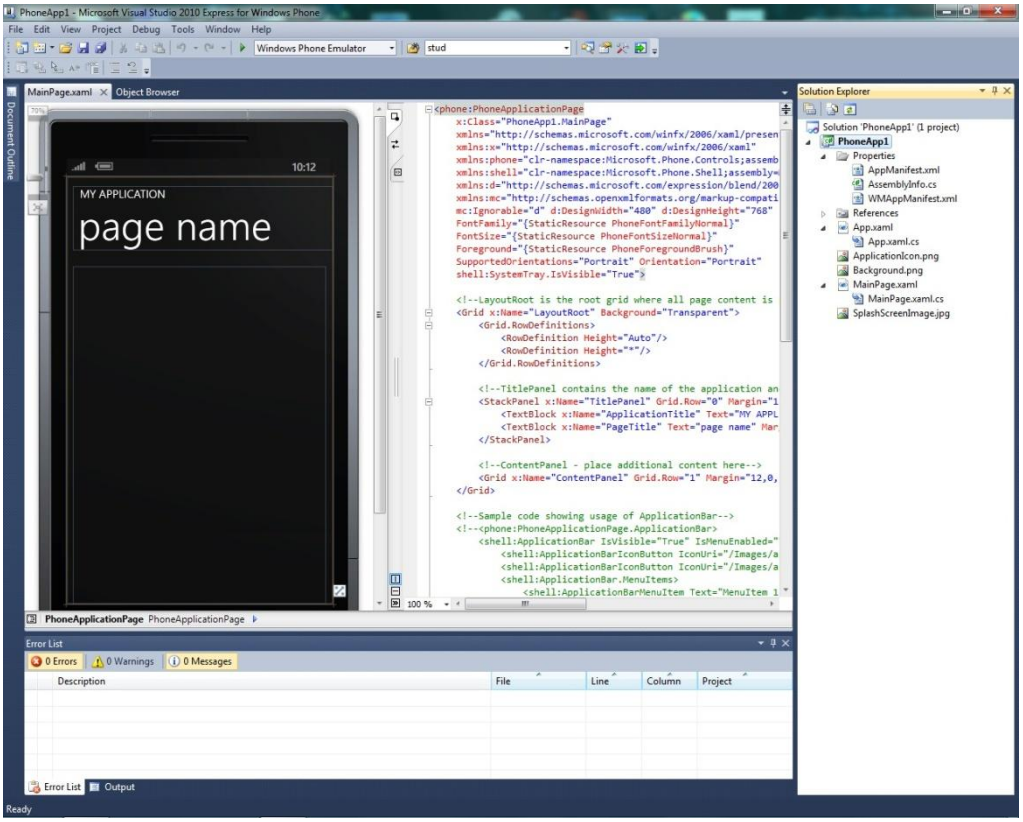
Windows PhoneScheduledTask Agent

- Implementuje algoritmy a aktivity, které se mají realizovat na pozadí, aniž by musela být hostitelská aplikace spuštěna.

Pro ukázku jsem vybral základní šablonu v SilverlightuWindows PhoneAplication.

Po vybrání šablony se otevře okno s možností výběru verze systému Windows Phone, pro který má být aplikace vytvořena. Na výběr máme z verzí 7.1 a 8.0. Vyberu si verzi 7.1, která je propagována jako verze 7.5(Mango).

Po zvolení šablony se objeví následující okno:



Obrázek 4- Visual Studio: základní zobrazení projektu

Pracovní plocha je rozdělena do několika částí. První část obsahuje telefon, na kterém se ukazuje, jak aplikace bude vypadat, dají se do ní jednoduše přidávat komponenty a měnit jejich vzhled. V prostřední části se nachází pracovní plocha pro tvorbu zdrojového kódu. Pokud, jako na obrázku č. 4, je otevřen XAML soubor, je plocha rozdělena na tyto dvě části - na XAML kód a aktuální vzhled telefonu. Pokud změníte vzhled v obrazovce telefonu, změní se kód v XAML souboru, pokud změníte XAML kód, změní se vzhled na obrazovce telefonu. Díky tomuto přístupu i bez hlubších znalostí dokážete měnit vzhled vašich aplikací. Ve třetí části se nachází struktura projektu. Každá aplikace je tvořena kořenovým souborem Solution. Tento soubor obsahuje kořenové projekty, které mohou být spolu provázány. Tučně vyznačený soubor je chápán jako startovní, ten je překládán a spouštěn.

Dále tu máme složku Properties, ta obsahuje tři soubory *AppManifest.xml*, *AssemblyInfo.cs*, *WMAppManifest.xml*. Všechny tyto soubory jsou generovány podle počátečního nastavení vlastností projektu. V určitých případech je vhodné to těchto souborů zasáhnout přímo. Především se jedná o soubor *WMAppManifest.xml*, který obsahuje parametry a specifikace pro publikování aplikace na MarketPlace.

Nejdůležitějšími soubory jsou*App.xaml* a *MainPage.xaml*, které přinášejí implementaci *App.xaml.cs* a *MainPage.xaml.cs*. *App.xml* a *App.xaml.cs* jsou hlavní soubory aplikace, které obsahují předgenerované metody v jazyce C#. Změny v těchto souborech jsou potřeba až tehdy, pokud chceme např. modifikovat chování hlavních metod a událostí aplikace.

Důležité soubory jsou pak také *MainPage.xaml* a *MainPage.xaml.cs*, které obsahují jak specifikaci vzhledu aplikace, tak aplikační logiku vázanou na události jednotlivých prvků.

Dále si můžeme všimnout několika obrázkových souborů. *ApplicationIcon.png*je obrázek ikony zobrazující se v seznamu aplikací. *Backgound.png* je obrázek, který je zobrazován jako pozadí aplikace. *SplashScreenImage.jpg* je obrázek který se zobrazuje ihned po spuštění aplikace.

7.2 Uživatelské rozhraní

V případě, že již máme vytvořený nový projekt ve Visual Studiu, automaticky se objeví první návrh aplikace, který je doplněn o prvky *Button*, *TextBlock*, *ListBox*, které jsou navzájem vnořeny do prvků *Grid*. Uživatelské rozhraní aplikace je postaveno na značkovacím jazyce s názvem Extensible

Obrázek 5- Uživatelské rozhraní

Applicaion MarkupLeanguage (XAML). XAML je formát XML, který umožňuje oddělit kód, který je použit na vzhled, od kódu, který je použit na funkčnost aplikace. Pokud umíte programovat v jazyce HTML, můžete pohlížet na XAML stejným způsobem, XAML se totiž také skládá z elementů a atributů. Pokud si chcete usnadnit práci a přitom neznáte tak dobře jazyk XAML, je ve Visual Studiu 2010 pro Windows Phone program nazývaný ExpresionBlend. Pomocí ExpresionBlendu můžete prvky pouze přidávat na uživatelské rozhraní a v nastavení měnit jeho vzhled a ExpresionBlend za vás vygeneruje příslušný XAML kód. Ale alespoň základní znalost XAML kódu je nutná, protože některé věci je potřeba upravit zásahem do XAML kódu. Pokud hledáte pomoc pro určitou položku, stačí umístit kurzor a stisknout klávesu F5 a zobrazí se nápověda. V souboru XAML na každý objekt UI prvek a u vlastnosti těchto objektů se nastavují atributy. XAML kód stejně jako u XML kódu vyjadřuje vztah jako dítě / rodič. Následující XAML kód ukazuje

StackPanek, který obsahuje obrázek a prvek *TextBlock*, jsou zde nastaveny i atributy.

```
<StackPanel Orientation="Horizontal" Margin="20"
Height="50"
VerticalAlignment="Top">
<Image Height="50" Width="50" Margin="20" Stretch="None"
Source="{Binding Picture}"/>
<TextBlock Text="{BindingName}"
VerticalAlignment="Right"/>
</StackPanel>
```

7.3 ApplicationBar

ApplicationBar je aplikační lišta sloužící pro zobrazení navigačních pomůcek. ApplicationBar je viditelný bez ohledu na to, zda je stránka posunutá či se na obrazovce zobrazí klávesnice. Jako ovládací prvky se používají *ApplicationBarIconButton* a *ApplicationBarMenuItem.*

Na aplikační liště můžeme mít jednotlivé akce vyjádřené pomocí ikon nebo jen za pomoci textu. Na obrázku vidíme rozšířenou podobu lišty, na které kde se nacházejí jak ikony, tak text. Ellipsis slouží k vysunutí a zasunutí aplikační lišty. **Obrázek 6 - Aplikační lišta**[24]

Chceme-li přidat aplikační lištu, tak vložíme do projektu prvek *ApplicationBar*. Následující XAML kód nám ukazuje nastavenou aplikační lištu, která je spojená z C# kódem na událost *Click*.

[24]Zdroj: http://www.diaryofaninja.com/blog/2011/07/05/solved-why-donrsquot-applicationbar-bindings-work-ndash-windows-phone-7-sdk

```
<phone:PhoneApplicationPage.ApplicationBar>
<shell:ApplicationBar IsVisible="True"
IsMenuEnabled="True">
<shell:ApplicationBarIconButton
IconUri="/obrazky/bakalarka.png"
Text="autor" Click="" IsEnabled="False" />
<shell:ApplicationBarIconButton
IconUri="/obrazky/bakalarka2.png"
Text="bakalarka detail" Click="tlacitko_Click" />
<shell:ApplicationBar.MenuItems>
<shell:ApplicationBarMenuItem Text="Menu"
Click="Menu_Click"/>
</shell:ApplicationBar.MenuItems>
</shell:ApplicationBar>
</phone:PhoneApplicationPage.ApplicationBar>
```

Je nutné nadefinovat, že máme v úmyslu použít aplikační lištu v naší aplikaci. Uděláme to pomocí *ApplicationBar*, který je součástí třídy *PhoneApplicationPage*. Dále je nutné nastavit potřebné atributy jako je *IsMenuEnabled* a *IsVisible.*ApplicationBar může obsahovat elementy *ApplicationBarIconButton* či *ApplicationBarMenuItem.ApplicationBarIconBut ton* by měl mít nastavenou hodnotu *IconUri.*Vlastnost Text určuje popis ikony.

Dále je potřeba přidat příslušné ikony do *ApplicationBaru*. Existuje několik ikon, které jsou součástí balíku Windows Phone SDK. Ikony lze nalézt v jednom z následujících umístění:

- Na 32-bitových operačních systémech: C: \ Program Files \ Microsoft SDK \ Windows Phone \ v7.1 \ Icons
- Na 64-bitových operačních systémech: C: \ Program Files (x86) \ Microsoft \ Windows Phone SDK \ v7.1 \ Icons

Pokud si chceme vytvořit vlastní ikony, musíme splnit následující požadavky:

- Ikona musí mít rozměr 48x48 pixelů.
- Ikona je centrována s velikostí 26x26 pixelů.

- Ikona by měla mít bílé pozadí s průhledností.
- Na aplikační liště by nemělo být více jak pět ikon.
- Ikona nemusí mít kolečko, lišta si sama kolečko vykreslí.

Chceme-li přidat ikonu do aplikační lišty, musíme obrázek nahrát do projektu. Následující XAML kód nám ukazuje, jakým způsobem přidáváme jednotlivé ikony do aplikační lišty.[12]

```
<shell:ApplicationBarIconButton
IconUri="/obrazky/ikona.png"
Text="popisek" Click="Tlacitko_Click" />
```

7.4 Přechod mezi stránkami, položkami

Přechody se obvykle konají v důsledku poklepáním na prvek v uživatelském rozhraní. Pro přechody nám ve Windows Phone slouží hypertextový odkaz, tlačítka nebo tlačítko zpět. Tlačítko zpět funguje bez jakéhokoliv našeho zásahu. Nejjednodušším způsobem, jak lze provádět přechod mezi stránkami, je přechod pomocí *HypperlinkButton.* Použijeme zde metodu *NavigationUri*, pomocí které přejdeme na požadovanou stránku. Příklad ukazuje, jak lze přejít na stránku z názvem Bakalarka.xaml.

```
<HyperlinkButton NavigateUri="Bakalarka.xaml" />
```

Pokud chceme používat *HypperlinkButton*, tak můžeme provádět navigaci pomocí třídy *NavigationService.* Tato třída obsahuje několik vlastností a metod, které nám pomohou s navigací. Následující kód nám ukáže, jak v C# po kliknutí na tlačítko, vyvoláme metodu pro přechod na požadovanou stránku.

```
Priváte void Tlacitko_Click(object sender, EventArgs e)
{NavigationService.Navigate(new
Uri("//složka/Bakalařka.xaml", UriKind.Relative));}
```

Metoda *Navigate*vždy vytváří novou instanci třídy, která má zadaný identifikátor URI. Pak zde máme tlačítko zpět, které vyvoláme metodou

NavigationService.GoBack, která vrací na předchozí stránku, ale jeto duplicita za tlačítko zpět na telefonu.[8]

7.5 Ovládací prvky

Ovládací prvky slouží k tomu, aby uživatel mohl zobrazovat data a komunikovat s aplikací. Máme zde na výběr mnoho prvků, jejichž užití je velmi jednoduché - stačí vybraný prvek „přetáhnout“ na pracovní plochu.

Prvky pro navigaci: *PhoneApplicationFrame, PhoneApplicationPage*

Layout prvky: *Border, Canvas, ContentControl, Grid, Panorama, Pivot, StackPanel, ScrollViewer, VirtualizingStackPanel*

Prvek pro zobrazení textu*: TextBlock*

Prvky pro vkládání textu: *TextBox, PasswordBox*

Prvek pro zobrazení seznamu položek: *ListBox*

Prvky pro tlačítka: *Button, HyperlinkButton*

Prvky pro výběr: *CheckBox, RadioButton, Slider, ComboBox*

Prvek pro zobrazení obrázku: *Image*

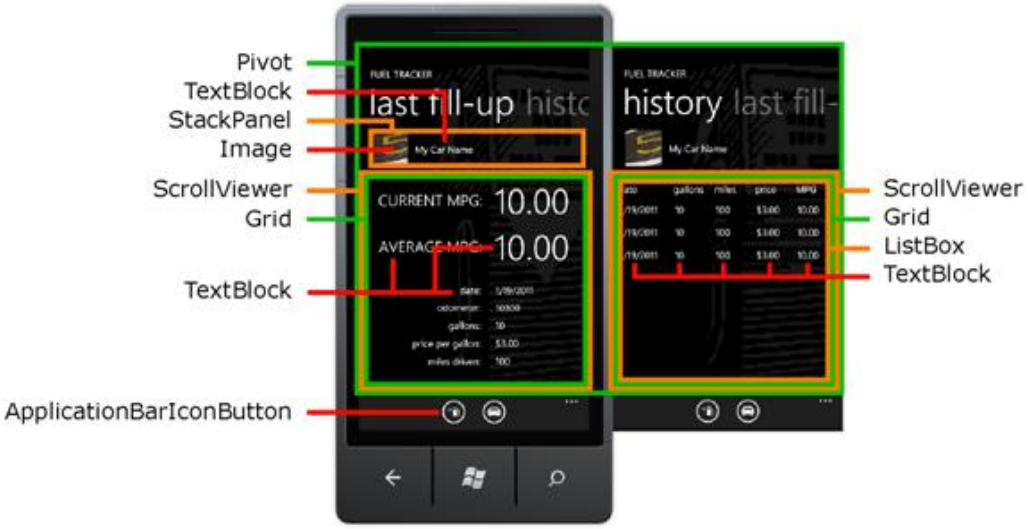
Prvek pro zobrazení mapy: *Map*

Prvek pro audio a video: *MediaElement*

Prvek pro HTML: *WebBrowser*

Prvek pro zobrazení načítání: *ProgressBar*

Prvek pro Pop-up zprávy: *Popup*



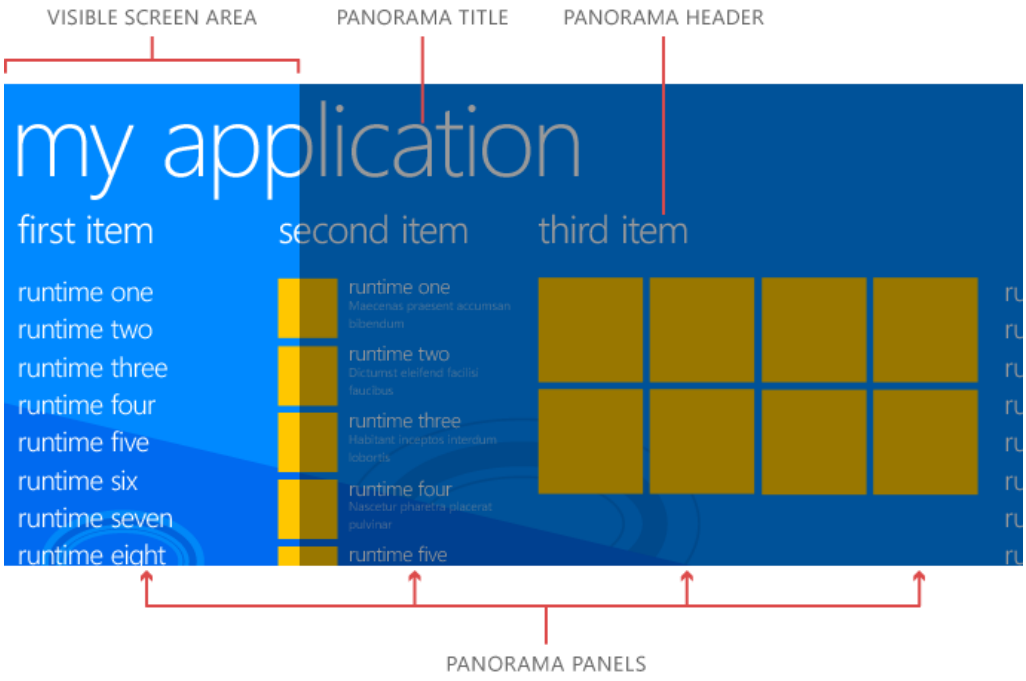
Obrázek 6- Ovládací prvky[25]

Na obrázku vidíme příklad použití některých ovládacích prvků. Lze si také všimnout, že je použit ovládací prvek Pivot, který nám rozšiřuje využitelný prostor aplikace. Operační systém Windows Phone má dva speciální ovládací prvky, které efektivně rozšiřují pracovní plochu telefonu. Jedná se o prvky Panorama a Pivot. Tyto prvky umožňují horizontální pohyb v obsahu telefonu.

[25]Zdroj: http://msdn.microsoft.com/en-us/library/gg680261(v=pandp.11).aspx

7.5.1 Panorama

Panorama prvek vytváří panoramatický vzhled položek, ve kterých jde listovat ze strany na stranu. Díky této možnosti se i značně odstraňuje omezená velikost displeje. Pokud chceme vytvořit projekt s panoramatickým vzhledem, stačí si najít příslušnou šablonu.



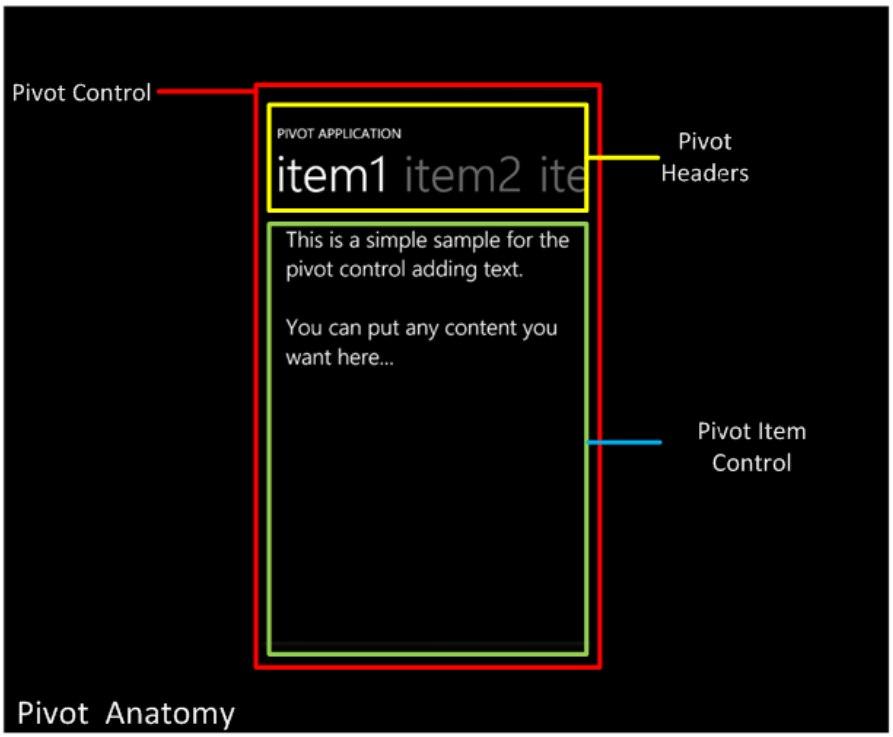
Obrázek 7- Panorama vzhled[26]

Prvek Panorama pochází ze jmenného prostoru Microsoft.Phone.Controls. Panorama prvek obsahuje atribut *Title*, což je nápis, který prochází všemi sekcemi. Každá sekce pak nese prvek *Header*. Dále obsahuje atribut *Background*, ve kterém je nastavené pozadí.

[26] Zdroj: http://msdn.microsoft.com/en-us/library/windowsphone/design/hh202912(v=vs.105).aspx

7.5.2 Pivot

Pivot prvek nabízí rychlý způsob, jak zobrazit velké množství dat v aplikaci. Může být použit i jako rozhraní pro navigaci mezi jednotlivými stránkami. Na rozdíl od panoramy, kde je určité překrytí dat, tak pivot prvek zaplňuje celou šířku stránky. Pokud chceme vytvořit projekt s Pivot vzhledem, stačí si najít příslušnou šablonu.



Obrázek 8- Pivot vzhled[27]

Prvek Pivot pochází ze jmenného prostoru *Microsoft.Phone.Controls*. Obsahuje atribut *Header*, který je titulkem dané sekce, dále obsahuje jeden až několik ControlItemů.

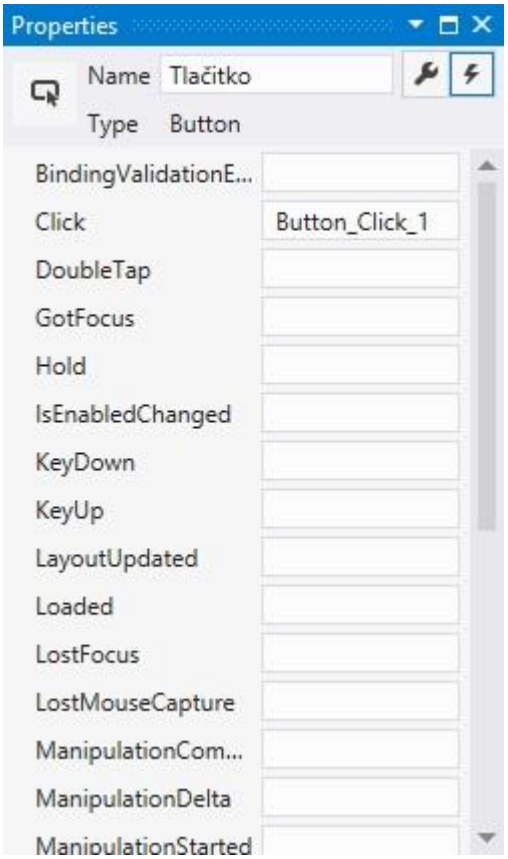
Dále zde máme prvky na zadávání textu. Přidáme-li ovládací prvky pro zadávání textu do aplikace, měli bychom povolit prostor pro klávesnici na obrazovce, musíme vyvolat panel, který by měl zajistit, že se klávesnice se

[27] Zdroj: http://msdn.microsoft.com/zh-cn/library/hh202919(v=vs.92).aspx

nebude vztahovat na tlačítka, nebo jiné ovládací prvky. Jedním způsobem, jak tomu zabránit, je umístění navigačních prvků v pruhu aplikace. Aby se zabránilo chybám, je třeba použít nejvhodnější klávesnice pomocí nastavením InputScope.

7.6 Reakce na ovládání událostí

Reakce na ovládání události nastane tehdy, pokud uživatel pracuje s obrazovkou, např. klepnutím, tažením nebo jinou manipulací s prvkem. To znamená, že události jsou postaveny na ovládacích prvcích uživatelského rozhraní. Například tlačítko je ovládací prvek a obsahuje událost Click, která je aktivní, pokud je tlačítko využito. Můžeme tedy vytvořit metodu pro zpracování události a to buď v okně s vlastnostmi, nebo v samotném XAML kódu. Okno s vlastnostmi obsahuje seznam všech akcí, které jsou k dispozici pro daný prvek. Na obrázku jsou uvedeny některé události pro prvek tlačítko.



Obrázek 9- Události pro tlačítko

Chceme-li vytvořit příslušnou událost na prvek tlačítko, stačí na prvek poklepat a událost se nám vytvoří. Otevře se příslušná metoda s kódem. Následující kódukazuje událost Click pro tlačítko.

```
Privatevoid PhotoButton_Click(object sender,
RoutedEventArgs e)
```

Můžeme také vytvořit příslušnou událost pomocí XAML kódu. Po přetáhnutí tlačítka se do XAML kódu vygeneruje kód tlačítka, který pak následně můžeme upravit. Následující obrázek nám ukazuje jak přidat v XAML kódu událost Click.

Obrázek 10- XAML kód tlačítka

Jakmile vyberete událost tlačítku, tak je potřeba nedefinovat příslušnou akci. Za Click dopíšeme znaménko = a objeví se nám nabídka, co vše může tlačítko ovládat.

Obrázek 11- XAML kód události

A následující kód ukazuje, jak je tlačítko spojeno s akcí Map_Click.[7]

```
<controls:PanoramaItemHeader="Bakalářka">
<Button
x:Name="Tlačitko"Content="Button"Click="Map_Click"
</controls:PanoramaItem>
```

7.7 Přidání obrázků a ikon

7.7.1 Obrázek v aplikaci

Chceme-li přidat obrázek do aplikace, musíme obrázek připojit k našemu projektu. Můžeme přidat obrázky pouze ve formátu JPG a PNG. Do projektu musíme přidat ovládací prvek *Image*. Dále v kódu nastavíme vlastnosti

obrázku. Následující XAML kód ukazuje, jak zobrazit obrázek s názvem bakalářka.jpg v naší aplikaci, který je uložen ve složce obrázky.

```
<Image Source="/obrazky/bakalarka.jpg">
```

Následující XAML kód nám ukazuje nastavení vlastností obrázku.

```
<Image Source="/obrazky/bakalarka.jpg"Stretch="None"
VerticalAlignment="Top"HorizontalAlignment="Left"
Margin="6,5,0,0"Height="207" />
```

7.7.2 Pozadí aplikace

Nyní si ukážeme, jak přidat obrázek, který bude sloužit jako pozadí aplikace. Při výběru obrázku na pozadí, bychom měli dodržovat některé zásady. Mezi ně patří především používání obrázků, které pracujíse světlým a tmavým tématem a v neposlední řadě používání obrázků ve formátu JPG nebo PNG. Pokud se chystáme nastavit průhlednost obrázku, měli bychom použít formát PNG.

Pro přidání obrázku na pozadí připojíme obrázek k projektu. Vytvoříme třídu *ImageBrush*a nastavíme cestu k obrázku pomocí *ImageSource*. Třída *ImageBrush*je deklarována v *App.xaml*,protože obrázek využívají všechny stránky aplikace.

```
<ImageBrush x:Key="plocha"
ImageSource="/obrazky/plocha.png" Stretch="None" />
```

Dále musíme aplikovat *ImageBrush* jako pozadí (v layoutu je umístěn veškerý obsah stránky).

```
<Gridx:Name="LayoutRoot" Background="{StaticResource
plocha}">
```

7.7.3 Úvodní obrázek

Úvodní obrazovka se nám ukazuje při každém spuštění aplikace. Ihned po vytvoření projektu se nám vytvoří i tento obrázek, který se nazývá *SplashScreenImage.jpg*. Chceme-li mít vlastní startovní plochu, stačí tento obrázek nahradit obrázkem o rozměrech 480x800 pixelů.

7.7.4 Ikona aplikace

Ikona aplikace je obrázek, který se zobrazuje v seznamu aplikací.

Chceme-li nastavit ikonu aplikace, musíme přidat obrázek do projektu a ve vlastnostech projektu nastavit ikonu aplikace.

Obrázek 12- Ikona aplikace

Popřípadě můžeme upravit prvek *IconPath*v souboru *WMAppMainfest.xml* a nastavit ikonu aplikace.

```
<IconPath IsRelative="true"
IsResource="false">KinIcon.png</IconPath>
```

7.7.5 Obrázek dlaždice

Obrázek dlaždice se zobrazuje tehdy, pokud si uživatel přidá vaší aplikaci na svou úvodní obrazovku.

Chceme-li přidat obrázek dlaždice, musíme obrázek nahrát do projektu. Pak, stejně jako u ikony aplikace, musíme zvolit obrázek ve vlastnostech projektu.

Obrázek 13- Obrázek na dlaždici

Popřípadě je opět možné upravit prvek *BlackgroungImageURI* v souboru *WMAppMainfest.xml* a nastavit obrázek dlaždice.[9]

```
<TemplateType>
<BackgroundImageURIIsRelative="true"
IsResource="false">KinIcon.png</BackgroundImageURI>
<Count>1</Count>
<Title>ikona</Title>
</TemplateType>
```

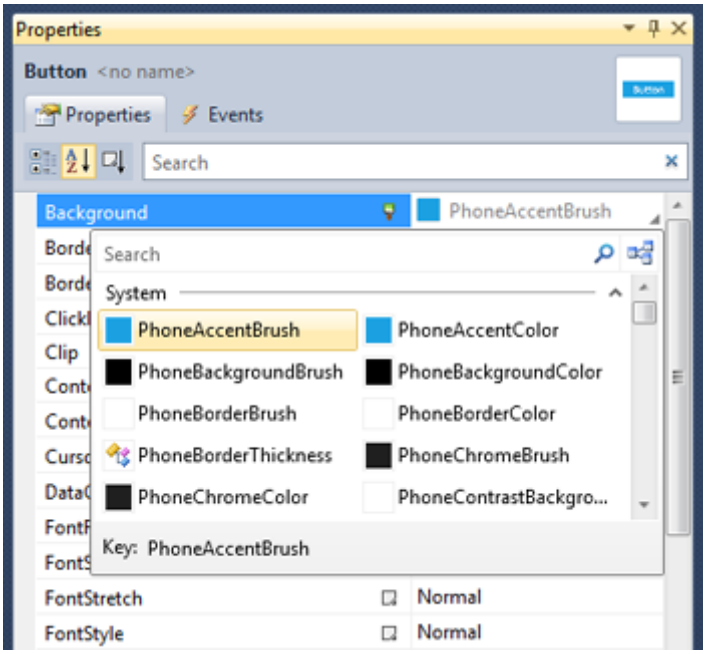
7.8 Barvy a témata

Operační systém Windows Phone využívá prvky témata. Každý uživatel si může pomocí nastavení upravit barvu dlaždic a kombinovat je s tmavým a světlým pozadím. Proto se u klasických aplikací nedoporučuje používat pevně dané barvy jednotlivých prvků. Místo toho se doporučuje u prvku využívat paletu *Resource*, což se ale netýká jen barev, ale i velikosti textu či fontu.

Máme na výběr z několika předdefinovaných stylů nebo si můžeme vytvořit svůj vlastní styl. U předdefinovaných stylů je podobný vztah jako mezi kaskádovými styly (CSS) a HTML, XAML kód nám umožňuje stejné vlastnosti.Pomocí stylů můžeme nastavit vzhled dat v aplikaci. Existuje mnoho předdefinovaných stylů, které jsou vhodné pro tmavé či světlé téma. Tyto styly zahrnují všechny barvy, fonty… Následující kód nám ukazuje, jakým způsobem nastavit tlačítko, aby když chceme změnit barvu, byla změna taková, jakou požadujeme.

```
<Button Content="Tlacitko" Height="50"
Background="{StaticResourcePhoneAccentBrush}" Width="130"
/>
```

Další možností je nastavení stylu tlačítka pomocí okna vlastnosti:[11]



Obrázek 14- Vlastnosti tlačítka[28]

7.9 Aplikační logika

Nyní, když jsme se seznámili s jazykem XAML a se specifikami vzhledu, je možné přistoupit k aplikační logice aplikace. Jedná se už pouze o programování kódu v jazyce C# ve Visual Studiu. Nejprve si ukážeme jednoduchý kód, který nám spustí Internet Explorer a námi zadanou stránku po stisknutí tlačítka.

```
void Tlacitko_MouseLeftButtonUp(object sender,
MouseButtonEvenArgs e)
        {
WebBrowserTask wbt = new WebBrowserTask();
```

[28]Zdroj: http://msdn.microsoft.com/en-us/library/gg680259(v=pandp.11).aspx

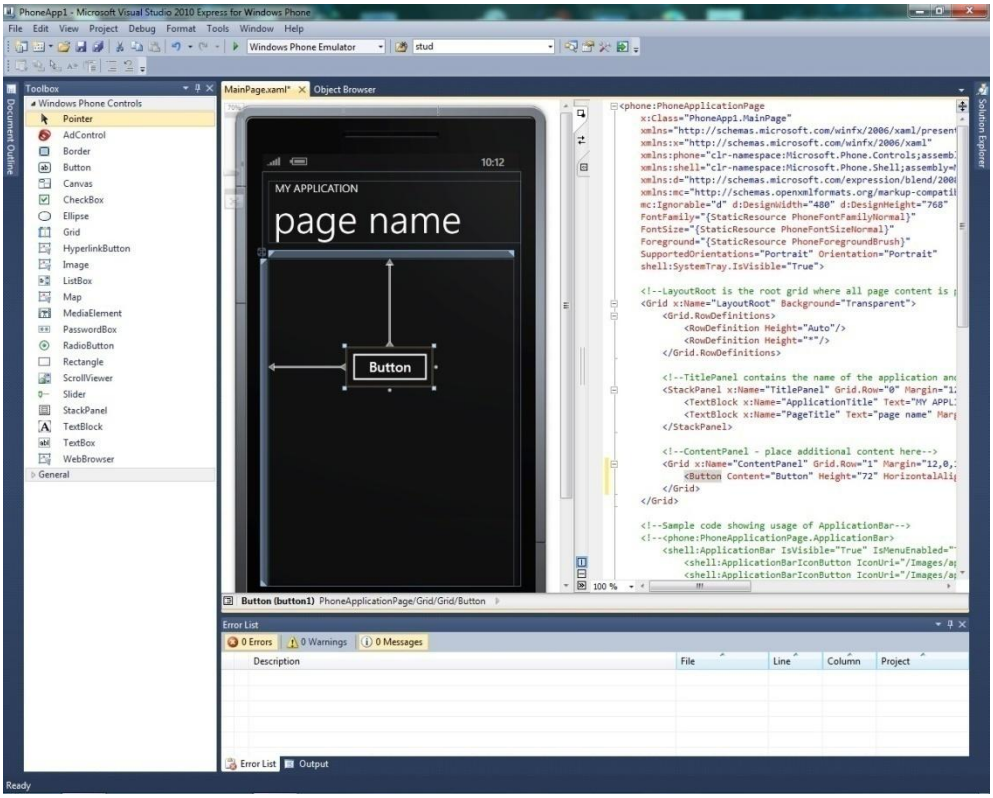
```
            wbt.URL = "http://www.pf.jcu.cz/ki/";
wbt.Show();
```

Dále si ukážeme, jak otevřít novou stránku v rámci aplikace. Tento kód je umístěn v souboru MainPage.xaml.cs. Převážně se tento přechod provádí pomocí metody NavigationService.Navigate.Navigace zpět na stránku se přenechává systému s využitím hardwarového tlačítka telefonu. Tuto funkci musí programátor zachovávat, protože její nedodržení bude mít za následek neschválení aplikace pro publikační portál MarketPlace.[10]

```
void MenuItem_Click(object sender, System.EventArgs e)
        {
NavigationService.Navigate(new
Uri(string.Format("/nova_stranka"), UriKind.Relative));
          }
```

8 První vytvořená aplikace

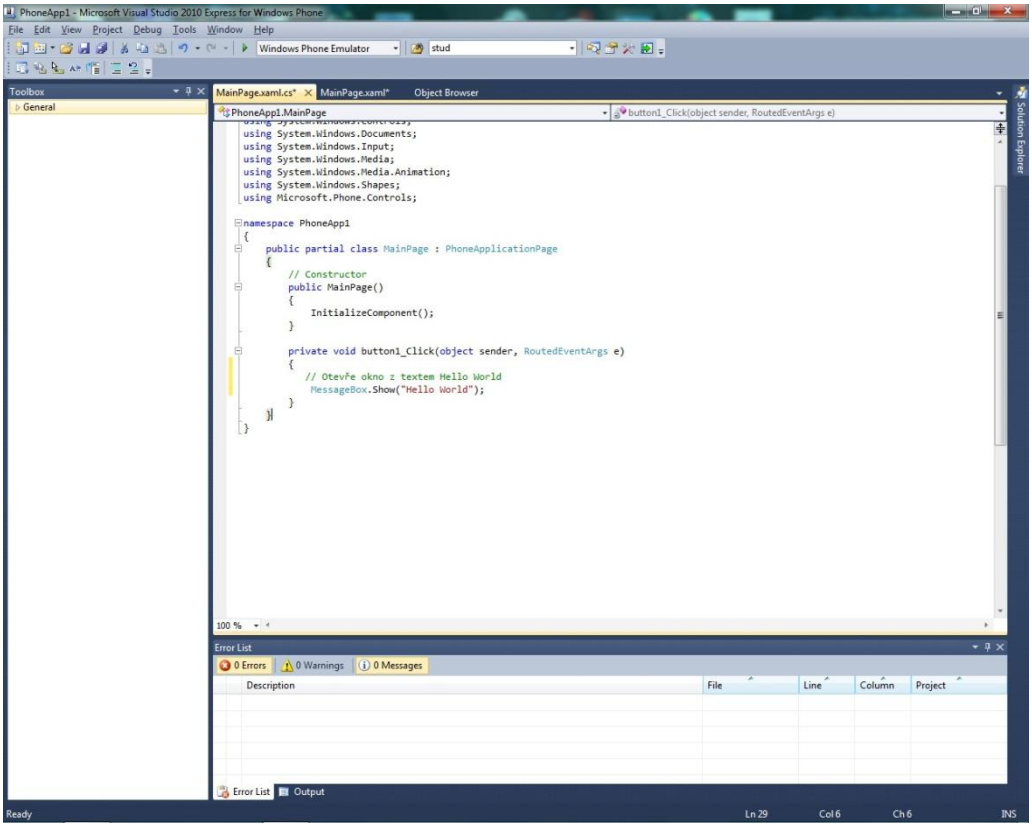
Teď, když máme představeny metody, funkce a vlastnosti operačního systému Windows Phone, zkusíme si naprogramovat klasickou aplikaci, se kterou začínají snad všichni lidé, kteří se učí programovat. Jedná se o klasické „Hello World“.



Obrázek 15- Visual Studio projekt (Hello Word)

V Toolboxu si vybereme prvek Button, který přemístíme na obrázek telefonu. Na pravé straně se přidá zdrojový kód (XAML) tlačítka. Zde si můžeme třeba jednoduše změnit název tlačítka. Pokud hledáme ještě snažší práci pro úpravu tlačítka, můžeme si projekt pustit v ExpressionBlendu. Dále je potřeba nadefinovat akci pro tlačítko, dvakrát na něj poklepáme a otevře se

nám okno z C# kódem.



Obrázek 16- Událost na tlačítko

Toto okno dobře zná každý, kdo už programoval v programovacím jazyce C#. Zde po klepnutí na tlačítko přiřadíme akci Message.Show(“Hello World“). Nyní můžeme aplikaci spustit (F5). Naběhne nám Emulátor telefonu a vněm se spustí naše aplikace. Klepneme na tlačítko a vyskočí okno s nápisem “Hello Word“. Právě máme hotovou naši první aplikaci.

9 Vývoj aplikace KIN

9.1 Příprava šablony pro aplikaci

Dále si v mé práci ukážeme naprogramování aplikace pro Katedru informatiky Pedagogické fakulty Jihočeské univerzity, která bude sloužit jako rozcestník pro pracovníky katedry, stávající studenty a uchazeče o studium. Studenti si nebudou muset na mobilu načítat stránky katedry, aby se dozvěděli hledané informace, postačí zapnutí aplikace a všechny potřebné informace budou mít ihned po ruce.

Jako první si musíme rozmyslet, jak by měla aplikace vypadat, zda použijeme šablonu Panorama či Pivot a v jaké technologii danou aplikaci naprogramujeme. Pro svou aplikaci zvolíme určitě technologii Silverlight, která se na tuto aplikaci optimálně hodí. Dále si ve Visual Studiu zvolíme šablonu Panorama, protože v aplikaci bude potřeba více stránek.

Obrázek 17- Výběr šablony

Posledním krokem je výběr verze systému, ve které chceme aplikaci vytvořit. Já si zvolím Windows Phone 7.1 který je zároveň i verzí 7,5.

Obrázek 18- Výběr operačního systému

Nyní máme před sebou obrazovku, kde máme šablonu z Panorama XAML kódem. Zde nalezneme dvě panorama stránky. Do naší aplikace si přidáme dalších pět stránek.

```
<controls:PanoramaItemHeader="jméno
stránky"></controls:PanoramaItem>
```

Naše stránky v aplikaci pojmenujeme: Aktuality, Kontakty, Najdete nás, Odkazy, Student, Katedra, Uchazeč. Dále je potřeba vymazat všechny nepotřebné komponenty v šabloně, jako jsou obrázky, *TextBoxy* atd. V souboru *WMAppManifest.xml* si nahrajeme své vlastní ikony pro aplikaci. Nyní máme zcela připravenou šablonu pro aplikaci amůžeme začít přidávat a programovat jednotlivé stránky a jejich vlastnosti.

9.2 Aktuality

Do aktualit budeme načítat data z webu katedry informatiky pomocí RSS. Nejprve si přidáme do šablony dvě komponenty TextBox a ListBox. TextBox budeme zobrazovat pouze do té doby, než se nám data načtou. ListBox nám bude sloužit k zobrazení aktualit z webu. Do třídy *MainViewModel.cs* přidáme *List*<*Link*>*NewsLinks*dále přidáme metodu *loadNews();* Nejdříve je potřeba si vytvořit metodu, která nám bude načítat odkaz RSS z xml souboru:

```
private void loadRss()
        {
StreamResourceInfoxml = Application.GetResourceStream(new
Uri("/MySchoolApp;component/Data/rss.xml",
UriKind.Relative));
XDocumentsettingsDoc = XDocument.Load(xml.Stream);

Settings = new Rss()
            {
Name = settingsDoc.Root.Element("name").Value,
NewsUrl = settingsDoc.Root.Element("newsUrl").Value,
                };

          }
```

Nyní si vytvoříme metodu loadNews(), která nám bude z parametru NewsUrl číst RSS data z webu Katedry Informatiky.

```
private void loadNews()
{
    LoadingState = MySchoolApp.LoadingState.LOADING;
    RaisePropertyChanged("LoadingState");


    if (NetworkInterface.GetIsNetworkAvailable() && !string.IsNullOrEmpty(App.ViewModel.Settings.NewsUrl))
    {
        string url = App.ViewModel.Settings.NewsUrl;

        var request = HttpWebRequest.Create(url);
        var result = (IAsyncResult)request.BeginGetResponse((iar) =>
        {
            try
            {
                var response = request.EndGetResponse(iar);

                using (var stream = response.GetResponseStream())
                using (var reader = new StreamReader(stream))
                {
                    string contents = reader.ReadToEnd();

                    Deployment.Current.Dispatcher.BeginInvoke(() =>
                    {
                        NewsLinks = Utils.GetLinksFromFeed(contents);

                        LoadingState = MySchoolApp.LoadingState.COMPLETED;
                        RaisePropertyChanged("LoadingState");
                        RaisePropertyChanged("NewsLinks");
                    });
                }
            }
            catch
            {
                LoadingState = LoadingState.ERROR;
                RaisePropertyChanged("LoadingState");
            }
        }, null);
    }
```

Obrázek 19 - Metoda pro RSS čtečku

9.3 Kontakty

V záložce kontakty budeme načítat všechny důležité kontakty, které jsou spojeny s Katedrou Informatiky. Na kontakt se bude dát kliknout a vyskočí na nás okno, ze kterého budeme moci rovnou volat nebo odesílat email. Do šablony si přidáme komponentu ListBox, ve které budeme zobrazovat načtené kontakty. Do třídy *MainViewModel.cs* přidáme *List<Contact>Contacts*dále přidáme metodu *loadContacts();*

Metoda na načítání kontaktů bude vypadat následovně:

```
Private void loadContacts()
        {
StreamResourceInfo xml =
Application.GetResourceStream(new
Uri("/KinApp;component/Data/Contacts.xml",
UriKind.Relative));
XDocumentcontactsDoc = XDocument.Load(xml.Stream);

Contacts = (from item in
contactsDoc.Descendants("contact")
Select new Contact
                          {
Name = item.Element("name").Value.ToString(),
                              Email =
item.Element("email").Value.ToString(),

PhoneNumber =
item.Element("phoneNumber").Value.ToString()
                          }).ToList<Contact>();
RaisePropertyChanged("Contacts");
```

Jak si můžeme všimnout, kontakty načítám z XML třídy (*Contact.xml*), kde načítám tři elementy a to: jméno, email a telefonní číslo. Struktura XML souboru vypadá následovně:

```
<contact>
<name>doc. PaedDr. Jiří Vaníček, Ph.D.</name>

<email>vanicek@pf.jcu.cz</email>
<phoneNumber>+420 387 773 079</phoneNumber>
</contact>
```

Následující kód nám nastaví situaci, kdy uživatel klikne na nějaký kontakt (změní se tedy výběr).

```
//MainPage.xaml.cs
Private void Contacts_SelectionChanged(objectsender,
SelectionChangedEventArgs e)
```

```
        {
if (e.AddedItems.Count == 0)
return;
var contact = e.AddedItems[0] as Contact;
if (!String.IsNullOrEmpty(contact.Email))
            {
if (!String.IsNullOrEmpty(contact.PhoneNumber))
                {
ShowContactSelection(contact);
                }
else
                {
EmailComposeTask ect = new EmailComposeTask();
ect.To = contact.Email;
ect.Show();}}
else
            {
if (!String.IsNullOrEmpty(contact.PhoneNumber))
                {
PhoneCallTask pct = new PhoneCallTask();
pct.PhoneNumber = contact.PhoneNumber;
pct.Show();
                }}
            ((ListBox)sender).SelectedItem = null;
```

Dále je potřeba naprogramovat události na klik pro jednotlivé elementy. Ukážeme si jak naprogramovat událost na klik pro email. Pro telefonní číslo je kód stejný jen se připíšou proměnné.

```
//MainPage.xaml.cs
private void ContactSelectionPhone_Click(object sender,
System.Windows.RoutedEventArgs e){
Var contact =
((FrameworkElement)sender).DataContextasContact;
PhoneCallTask pct = new PhoneCallTask();
pct.PhoneNumber = contact.PhoneNumber;
pct.Show();
        }
```

9.4 Najdete nás

V záložce najdete nás, se bude načítat obrázek mapy, který bude nahrán v aplikaci, plus souřadnice polohy školy. Do šablony si dáme komponentu tlačítko, kterému dáme textovou a grafickou podobu a bude nás odkazovat na obrázek, který se nám bude otevírat na další stránce (*MapPage.xaml*). Do *MainPage.xaml.cs* přidáme událost na klik pro tlačítko, které nás bude odkazovat na *MapPage.xaml* třídu.

```
private void CampusMap_Click(object sender,
RoutedEventArgs e)
        {
NavigationService.Navigate(new Uri("/MapPage.xaml",
UriKind.Relative));

        }
```

Ve třídě *MapPage.xaml* přidáváme Layout, ve kterém je náš obrázek s nastavenými hodnotami.

```
<Image x:Name="MyImage" Source="Resources/map.jpg">
```

Ve třídě *MapPage.xaml.cs* nastavujeme přibližování, oddalovaní obrázku a převracení obrázku podle polohy telefonu.

9.5 Odkazy

V záložce odkazy nalezneme všechny důležité odkazy, které souvisejí s katedrou informatiky. Tyto odkazy se budou otevírat v prohlížeči Internet Explorer. Do šablony si přidáme ListBox, ve kterém se nám odkazy budou zobrazovat. Do třídy *MainViewModel.cs* přidáme *List<Link>Links*, dále přidáme metodu *loadLinks();* Tato metoda načítá třídu *Links.xml*, ve které jsou uloženy atributy s odkazy.

```
private void loadLinks(){
Links = parseLinkFile("/Data/Links.xml");
RaisePropertyChanged("Links");

        }
```

Dále musíme naprogramovat metodu, která nám odkazy z XML souboru přečte.

```
private List<Link>parseLinkFile(string ressource Path)
        {
StreamResourceInfo xml =
Application.GetResourceStream(new
Uri("/MySchoolApp;component" + resourcePath,
UriKind.Relative));
return (from item in
XElement.Load(xml.Stream).Elements("link")
select new Link{
Title = item.Element("title").Value,
Url = item.Element("url").Value,
AditionalInfo = linksParseHelper(item.Element("bool")),
                    }).ToList<Link>();
        }
```

Struktura načítaného XML souboru vypadá následovně:

```
<link>
<url>https://www.facebook.com/pages/Katedra-informatiky-
PF-JU/154029028034826</url>
<title>Kin na Facebooku</title>
</link>
```

9.6 Student, Katedra, Uchazeč

Tyto záložky budou totožné, jen každá záložka bude na své Panorama stránce a načítat svá data ze svého přiřazeného XML souboru. Tyto záložky se budou stahovat přímo z webové stránky Katedry Informatiky. Na tyto záložky použijeme komponentu *WebView*, do které se budou načítat data z webu KIN. Do šablony si opět přidáme ListBox, do kterého budeme načítat položky z XML souboru.*PanoramaItem* bude vypadat následovně:

```
<controls:PanoramaItemHeader="Student">
<ListBox x:Name="ForStudentsListBox"
          Margin="12,0,0,0"
```

```
      SelectionChanged="NavigateToPageUrlListBox_Selection
Changed"
ItemsSource="{BindingForStudents}"
      ItemTemplate="{StaticResourceLinkDataTemplate}"/>
</controls:PanoramaItem>
```

V Headeru si nastavíme nadpis záložky, ten přidáme do */Localization/AppResources.resx*). Dále máme název našeho *ListBoxu*. *SelectionChanged*podle toho, na co kliknu (a toho, s čím to mám pomocí *ImtesSource* svázané), dohledá objekt z příslušné *List<Link>* kolekce a odnaviguje se na jeho proměnnou URL (dříve načtenou z XML souboru). *Binding*, který nás odkazuje na metodu *ForStudent.* A jako poslední se zde nachází *ItemTemplate,*ve kterém je nastavený vzhled.

Do třídy *MainViewModel.cs* přidáme *List<ForStudent>ForStudents*dále přidáme metodu *loadForStudents();*

```
privatevoidloadForStudents()
        {
ForStudentsLinks =
parseLinkFile("/Data/ForStudents.xml");
RaisePropertyChanged("ForStudents");

          }
```

Tato metoda nám načte XML soubor (ForStudents.xml), ve kterém máme odkazy na jednotlivé záložky. Struktura XML souboru vypadá následovně:

```
<link>
<url><![CDATA[/UrlWebPage.xaml?title=Změny ve
výuce&page=http://wvc.pf.jcu.cz/ki/?article=/zmeny-ve-
vyuce.html]]></url>
<title>Změny ve výuce</title>
<bool>False</bool>

</link>
```

Do title zadáme text, který chceme mít v dané *PivotPage*. Do url zadáme odkaz na stránku Katedry Informatiky.

A jako nejdůležitější a hlavní částí aplikace zbývá naprogramovat zobrazování informací z webu Katedry Informatiky v aplikaci.

Nejprve si vytvoříme *Pivot PageUrlWebPage.xml*, která bude sloužit pro všechna zobrazovaná data. Zde jako hlavní komponenta bude sloužit tzv. *WebView*. U WindowsPhone najdete tuto komponentu pod názvem *WebBrowser*.

```
<Grid
x:Name="ContentPanel"Grid.Row="1"Margin="12,0,12,0">
<phone:WebBrowserVisibility="Collapsed"
cxi:WebBrowserHelper.Html="{BindingUrlWeb.TheTextItself}"
x:Name="webBRWSR"HorizontalAlignment="Left"VerticalAlignm
ent="Top"Height="700"Width="456"/></Grid>
```

Nyní je potřeba naprogramovat WebBrowser(WebView), který nám bude zobrazovat Html řetězec. Zde zvolíme metodu NavigateToString, která nám bude procházet Html řetězec a ten zobrazovat.

```
public static class WebBrowserHelper
{
    public static readonly DependencyProperty HtmlProperty = DependencyProperty.RegisterAttached(
        "Html", typeof(string), typeof(WebBrowserHelper), new PropertyMetadata(OnHtmlChanged));

    public static string GetHtml(DependencyObject dependencyObject)
    {
        return (string)dependencyObject.GetValue(HtmlProperty);
    }

    public static void SetHtml(DependencyObject dependencyObject, string value)
    {
        dependencyObject.SetValue(HtmlProperty, value);
    }

    private static void OnHtmlChanged(DependencyObject d, DependencyPropertyChangedEventArgs e)
    {
        var browser = d as WebBrowser;
        if (browser == null)
            return;

        var html = e.NewValue.ToString();
        browser.NavigateToString(html);
    }
}
```

Obrázek 20 - Metoda pro zobrazení html řetězce

Dále na stránce můžeme narazit na odkazy,které budou odkazovat mimo stránky katedry. Tyto stránky budeme otevírat mimo aplikaci v IE. Na stránkách se můžeme také setkat s dokumenty jako PDF či DOC. K tomuto nám bude sloužit následující metoda:

```
void webBRWSR_Navigating(object sender, NavigatingEventArgs e)
{
    if (!viewMod.BoolDeleteLinks)
    {
        e.Cancel = true;
        if (e.Uri.Host.Equals("wvc.pf.jcu.cz") && !e.Uri.AbsolutePath.Contains(".pdf")
            && !e.Uri.AbsolutePath.Contains(".doc"))
        {
            NavigationService.Navigate(new Uri(String.Format("/UrlWebPage.xaml?title={0}&addInfo={1}&page={2}",
                string.Empty, false, e.Uri), UriKind.Relative));
        }
        else
        {
            var wbt = new WebBrowserTask();
            wbt.Uri = e.Uri;
            wbt.Show();
        }
    }
}
```

Obrázek 21 - Metoda pro navigaci v aplikaci

Nyní bude potřeba udělat nový model, kde webovou stránku stáhneme a následně jí budeme upravovat, aby se nám zobrazovala ona potřebná data. Vytvoříme si tedy třídu *UrlWebViewModel.cs*která nám bude sloužit pro všechny zobrazované položky. K této třídě si přiřadíme i příslušnou *pivot page.*

Nyní si vytvoříme metodu, která nám stáhne celý zdrojový kód načítané stránky.

```
void client_DownloadStringCompleted(object sender, DownloadStringCompletedEventArgs e)
{
    if (e.Error != null)
        return;
    var r = e.Result;
    var b = r.Split(new string[] {"<div class=\"line-undercontent\">", "</div>"}, StringSplitOptions.RemoveEmptyEntries);
    var c = b[28];

    string _caption = string.Empty;
    string _bodyString = prefix + c + postfix;

    _bodyString = fastConvertExtendedASCII(_bodyString);
    if (BoolDeleteLinks) { _bodyString = deleteLinks(_bodyString); }

    UrlWeb.TheTextItself = _bodyString;
    RaisePropertyChanged("UrlWeb");
}
```

Obrázek 22 - Metoda pro uložení zdrojového kódu

Zdrojový kód stránky je pak potřeba zpracovat a dekódovat. Vytvoříme si metodu pro získání znaků ze zdrojového kódu.

```
private string fastConvertExtendedASCII(string HTML)
{
    char[] s = HTML.ToCharArray();



    int n = 0;
    int value;
    foreach (char c in s)
    {
        if ((value = Convert.ToInt32(c)) > 127)
        {
            if (value > 9999)
                n += 7;
            else if (value > 999)
                n += 6;
            else
                n += 5;
        }
    }
```

Pro konečný (zobrazený) zdrojový kód je potřeba přiřadit vyrovnávací paměť.

```
char[] res = newchar[HTML.Length + n];
```

Nyní bude potřeba vytvořit metodu, která nám vybere potřebná data a vrátí nám potřebný *string*.

```
int i = 0;
int div;
const int zero = (int)'0';
foreach (char c in s)
{
    if ((value = Convert.ToInt32(c)) > 127)
    {
        res[i++] = '&';
        res[i++] = '#';

        if (value > 9999)
            div = 10000;
        else if (value > 999)
            div = 1000;
        else
            div = 100;

        while (div > 0)
        {
            res[i++] = (char)(zero + value / div);
            value %= div;
            div /= 10;
        }

        res[i++] = ';';
    }
    else
    {
        res[i] = c;
        i++;
    }
}

return new string(res);
```

Obrázek 23 - Metoda pro konverzi textu

A jako poslední přidáme *string prefix* (hlavičku), který budeme přidávat ke zdrojovému kódu stránkya podle kterého se bude formátovat celý text v aplikaci. *String* bude obsahovat klasické Html tagy (v UTF-8). Vytvoříme si dva *stringy*, jeden pro tmavé téma a druhý pro světlé téma, a podle nastavení telefonu se nám aktivuje příslušný *string*.

```
Public bool Black { set { if (value) { prefix =
prefixBlack; } else { prefix = prefixWhite; } } }

string postfix = "</body>";
string prefix = string.Empty;
string prefixWhite = "<!doctype><meta name=\"Viewport\"
content=\"width=400; user-scaleable=no; initial-
scale=1.0\"><body style=\"color: #000; background-color:
#fff;\"link=\"orange\"; alink=\"orange\";
vlink=\"orange\";><body>";
string prefixBlack = "<!doctype><meta name=\"Viewport\"
content=\"width=400; user-scaleable=no; initial-
scale=1.0\"><body style=\"color: #fff; background-color:
#000;\" link=\"orange\"; alink=\"orange\";
vlink=\"orange\";><body>";
```

9.7 Souhrn funkčnosti aplikace

Aplikace se zapne, načte se hlavní stránka, v *MainModelu* se z XML souborů načtou linky a případně i texty. Jakmile uživatel klikne na konkrétní odkaz, stáhne se celý zdrojový kód dané stránky (ve *UrlWebModel*), ořízne se o menu atd. a zobrazí se v klasickém *WebView*. Navíc se ke kódu přidá hlavička s vlastním formátováním.

10 Testování a ladění aplikace

Součástí každého vývoje aplikací je testování a ladění. Díky průběžnému testování a ladění se můžeme vyhnout problémům při certifikaci aplikace. U Windows Phone máme dva způsoby, jak můžeme aplikace testovat, a to buď na emulátoru, nebo na fyzickém zařízení.

10.1 Windows Phone Emulátor

Windows Phone Emulátor je jednou z možností, jak otestovat vytvořenou aplikaci. Emulátor je součástí balíku SDK. Díky tomuto emulátoru můžeme začít s vývojem aplikace na Windows Phone, aniž bychom vlastnili telefon s tímto operačním systémem. Emulátor simuluje poslední operační systém, který mají Windows Phone telefony, což umožňuje testování základních vlastností a možností celé platformy. Problémem jsou ale omezené možnosti - např. nejde uzamykání obrazovky, nefunguje plnohodnotný zoom na obrázky či internetové stránky a nastavení telefonu je poněkud omezené. Pokud chceme testovat aplikace v emulátoru, musíme si ve Visual Studiu nastavit Windows Phone Emulator. Emulátor spustíme klasickou zkratkou F5. Jakmile spustíme emulátor, tak se nám aplikace do emulátoru nainstaluje a spustí. Na boční pravé straně emulátoru nalezneme lištu s tlačítky,

Obrázek 24 - Emulátor

která slouží k otáčení telefonu a otestování, zda se nám aplikace překlápí. Také je zde možné virtuálně pohybovat telefonem v prostoru a simulovat akcelerometr či gyroskop nebo také simulovat GPS souřadnice.

10.2 Fyzické zařízení

Ačkoliv je emulátor velice kvalitní nástroj, nedokáže plnohodnotně nahradit fyzické zařízení. Některé funkce je zkrátka potřeba otestovat na reálném zařízení. Také je potřeba otestovat skutečný výkon a plynulost běhu aplikace, protože v případě emulátoru mohou být informace výrazným způsobem zkreslené.

Testování na reálném zařízení vyžaduje určité požadavky. Mezi tyto požadavky patří:

- Musíte mít platný vývojářský účet (buď studentský nebo zaplacený roční poplatek 99$).
- Vlastnit telefon s operačním systémem Windows Phone a mít nainstalovaný Microsoft Zune.

Předtím, než budete nahrávat aplikace na reálné zařízení, je potřeba telefon zaregistrovat a odemknout pro vývojové účely. Odemknutí telefonu se provádí pomocí Microsoft Zune. Pokud máte svůj telefon připravený, to znamená: odemčený, připojený USB k počítači, zapnuté Zune a telefon spuštěný v hlavní nabídce, můžete začít testovat aplikace na svém telefonu. Ve Visual Studiu zvolíme Windows Phone Device a stiskneme F5, poté stejně jako v emulátoru se nám aplikace do telefonu nainstaluje a spustí.

11 Publikování aplikace na MarketPlace

MarketPlaceje portál sloužící k tomu, abychom si mohli oficiálně stáhnout aplikace do telefonu, ať už přímo z telefonu, pomocí desktopové aplikace Zune nebo webové varianty www.windowsphone.com.

Pro publikování naší aplikace navštívíme stránku dev.windowsphone.com, zde klikneme na tlačítko Submit APP.

V záložce App info vyplníme název aplikace, který se bude zobrazovat na webu v APP HUB[29] systému, nikoliv v MarketPlace. Vyplníme zde také číslo verze, dále si vybereme kategorii, do které chceme svou aplikaci vložit a následně klikneme na tlačítko Save.

V záložce Upload and describe your XAP[30] vybereme náš soubor XAP, který budeme chtít nahrát. Vyplníme popis aplikace, který se bude zobrazovat v MarketPlace a klíčová slova, díky nimž je možné aplikaci v MarketPlace vyhledat.

Dalším nutným krokem je příprava doplňkových zdrojů.

- **Small mobile app tile**

  Obrázek ve formátu PNG o rozměru 99x99 pixelů. Tento obrázek je použit v aplikaci na MarketPlace.

- **Large PC app tile**

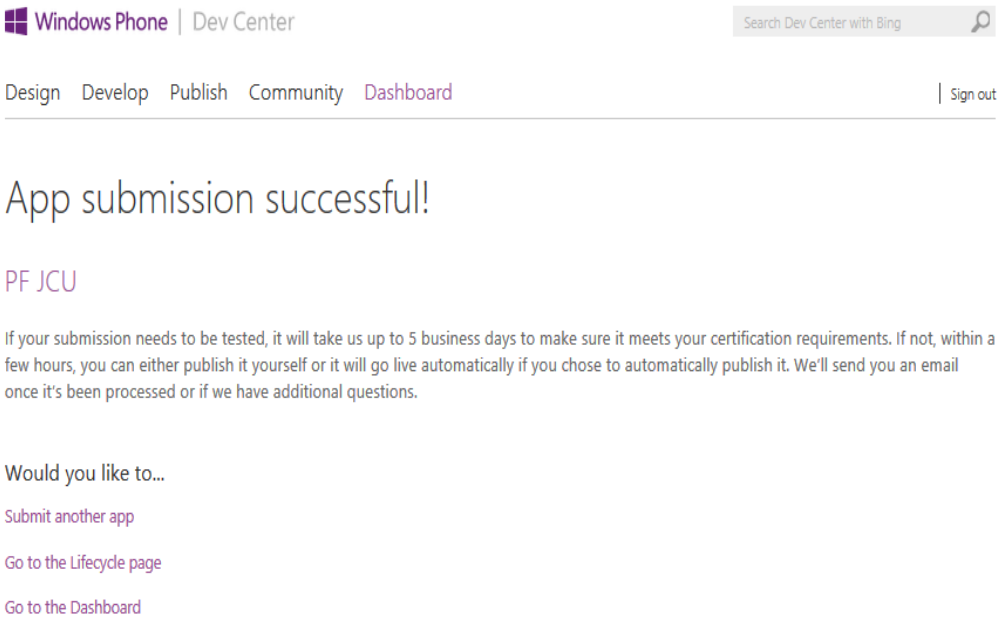
  Obrázek ve formátu PNG o rozměrech 200x200 pixelů. Tento obrázek je použit v aplikaci Zune.

[29] Zdroj: https://dev.windowsphone.com/en-us/publish
[30] Hlavní soubor aplikace

- **In appscreenshots**

  Obrázky ve formátu PNG o rozměrech 480x800 pixelů. Tyto obrázky slouží jako náhled na aplikaci před stažením. Musíme nahrát alespoň jeden obrázek a maximálně osm obrázků. Nejsnadnější cesta jak screenshoty vytvořit je pomocí Windows Emulátoru, který zachycuje přímo otevřenou plochu a ukládá screenshoty přímo ve formátu PNG a velikosti 480x800 pixelů.

V posledním kroku klikneme na tlačítko Review and submit, kde si zkontrolujeme vyplněné údaje a stiskem tlačítka odešleme aplikaci k certifikaci.



Obrázek 25 - Úspěšné nahrání aplikace

11.1 Certifikace

Certifikace probíhá tak, že skupina testerů otestuje zaslanou aplikaci na různých zařízeních. Certifikace většinou proběhne do pěti pracovních dnů.

Pokud certifikace proběhne úspěšně, přijde vám email z gratulací a vaše aplikace bude umístěna na MarketPlace. Můžeme zde také pouze aktualizovat naší schválenou aplikaci. Aktualizace je připravena na MarketPlace ke stažení hned druhý pracovní den.

Může ale také nastat situace, kdy aplikace neprojde certifikací. V tomto případě vám přijde email s informací, kde naleznete, že aplikaci nebylo možné certifikovat. Zprávu o problému při certifikaci najdete na APP HUBU v záložce My Dashboard. Ve zprávě naleznete detailní popis problému a vypsané důvody, proč vaše aplikace nemohla být certifikována. Dále je zde také uveden a popsán postup směřující k nápravě problému.

Windows® Phone Marketplace

Certification Test Results

Application Details

Name: PF JCU
Version: 1.0.0.0
Company Name: Striker
Windows Phone OS Version: 7.1
Test ID: 551149

Submission Received: 05/13/2013
Testing Completed: 05/16/2013

Application Test Details

Capabilities Tested: Networking

Language(s): Czech

Result: Failed
Failure Summary: 5.5.2
Exception(s) Applied: None

Action: Please address the comprehensive list of failures below, review the Windows Phone Application Certification requirements (http://go.microsoft.com/fwlink/?LinkID=183220) and resubmit your updated application for certification testing. For further assistance, please submit a support ticket using the Support e-Form in the Dev Center Dashboard (http://go.microsoft.com/?linkid=9782121).

Windows Phones Tested: HTC 8S, HTC 8X / 6990LVW , HTC Titan, Nokia Lumia 610, Samsung Focus Flash

Technical

5.5 Content Validation

5.5.2

Requirements

App content, such as text and visual elements, must be visible and legible regardless of the phone theme without panning or zooming. For example, if the phone theme changes from black background to white background, the text and visual elements of your app must be visible or legible. As another example, when content loads into a WebBrowsercontrol, the text and visual elements must be visible or legible without requiring the user to pan or zoom.

Expected Result

1. Navigate to the Settings page in the app list.
2.Tap theme and change Background to 'Dark'.
3.Launch your app.
4.Verify that the text and visual elements of the app are visible and legible without requiring the user to pan horizontally or zoom.
5.Navigate back to the theme page under Settings, and change Background to 'Light'
6.Launch your app.
7.Verify that the text and visual elements of the app are visible and legible without requiring the user to pan horizontally or zoom.

Comments: When the device's theme is set to light, the textual content of the application cannot be seen. **Result:** Fail

Steps to Reproduce:
1. Set the device's theme to light.
2. Launch the application.
3. Observe the application's content cannot be seen.

Obrázek 26 - Popsání problému proč aplikace nebyla certifikovaná.

Na screenu vidíme, že se vyskytl problém se světlým tématem, že není aplikace viditelná. Po opravení problémů je potřeba opět nahrát soubor XAP

a podstoupit certifikační proces. Pokud jsme opravili vše podle instrukcí, které byly popsané na obrázku výše, mělo by bez problémů dojít k certifikování aplikace.

Obrázek 27 - Ocertifikování aplikace

Ihned poté, co byla aplikace certifikována, se objeví na MarketPlace. Dále na webu APP HUB máte k dispozici průběžné statistiky stažení aplikace, záznamy o pádech aplikace, zde je napsáno proč a při čem aplikace byla náhle ukončena. Také zde nalezneme ekonomické výsledky placených aplikací. Všechny tyto informace jsou aktualizovány se zpožděním několika dní.

Dále můžeme naši aplikaci aktualizovat nahrazením novější verzí. Stačí pouze nahrát XAP soubor, vyplnit číslo verze a pokud všechny předešlé informace

souhlasí, můžeme aplikaci odeslat k certifikování. Certifikace aktualizace trvá pouze jeden pracovní den.

12 Závěr

V mé bakalářské práci jsem měl za úkol seznámit čtenáře s vývojem aplikací na poměrně nový mobilní operační systém Windows Phone.

V mé práci předpokládám základní znalosti vývojového prostředí Visual Studio a programovacího jazyka C#. Popsal jsem zde všechny technologie pro vývoj aplikací a spíše jsem se zaměřil na technologii Silverlight, která slouží k programování aplikaci, kdežto technologie XNA je spíše pro graficky náročné aplikace (hry). Z technologie Silverlight pak vychází i má praktická část. V práci jsou popsány všechny důležité komponenty, které programátor potřebuje k tomu, aby udělal jakoukoliv aplikaci. Dále v mé práci popisuji praktickou část, ve které se zabývám naprogramovanou aplikací pro Katedru informatiky Pedagogické fakulty Jihočeské univerzity, která slouží jako rozcestník pro pracovníky katedry, studenty a uchazeče o studium. V této částijsem přidal a popsal zdrojové kódy všech důležitých metod, pomocí kterých aplikace funguje. Aplikace je dostupná ke stažení na mobilním MarketPlace nebo na adrese: http://www.windowsphone.com/en-us/store/app/kin-jcu/a97940e7-91bb-4623-b182-2ff74249dd1c, kde je možné si jí nechat do telefonu odeslat.

Stanovené cíle bakalářské práce byly splněny.

13 Seznam použité literatury a zdrojů

[1] OS Windows Mobile/Phone: strmá cesta historií. In: SMRČEK, Jakub. *Cnews.cz* [online]. 1.květen 2011. cnews, 2011, 1.květen 2011 [cit.2013-05-10]. Dostupné z: http://www.cnews.cz/os-windows-mobilephone-strma-cesta-historii

[2]Windows Phone: Windows Phone 7 updatehistory. Windows Phone [online]. 2013 [cit. 2013-06-11]. Dostupné z: http://www.windowsphone.com/en-us/how-to/wp7/basics/update-history

[3]WMpoweruser: Microsoft relaxes WP7 hardware requirements. . *WMpoweruser* [online]. 2011 [cit. 2013-06-11]. Dostupné z: http://wmpoweruser.com/microsoft-relaxes-wp7-hardware-requirements-enable-camera-less-phones/

[4]WhatIsthe Microsoft XNA Framework?. CASTEELE, John. DEMAND MEDIA. Azcentral: ArizonasHomePage [online]. 2013 [cit. 2013-05-11]. Dostupné z: http://yourbusiness.azcentral.com/microsoft-xna-framework-20700.html

[5]Microsoft. In: *Microsoft* [online]. Microsoft, 2013 [cit. 2013-06-10]. Dostupné z: http://www.microsoft.com/cze/web/silverlight/windows-phone-7.aspx

[6]Windows Phone SDK 7.1. MICROSOFT. Microsoft: Download Center [online]. 25.9.2011. 2011 [cit. 2013-05-10]. Dostupné z: http://www.microsoft.com/en-us/download/details.aspx?id=27570

[7]MSDN: Microsoft. In: *MSDN: Microsoft* [online]. Microsoft, 2011, 21.10.2011 [cit. 2013-06-10]. Dostupné z: http://msdn.microsoft.com/en-us/library/gg680261(v=pandp.11).aspx

[8]MSDN: Microsoft. In: *MSDN: Microsoft* [online]. Microsoft, 2011, 21.10.2011 [cit. 2013-06-10]. Dostupné z: http://msdn.microsoft.com/en-us/library/gg680263(v=pandp.11).aspx

[9]MSDN: Microsoft. In: *MSDN: Microsoft* [online]. Microsoft, 2011, 21.10.2011 [cit. 2013-06-10]. Dostupné z: http://msdn.microsoft.com/en-us/library/gg680265(v=pandp.11).aspx

[10]Windows Phone: výukový kurz. *Windows Phone: výukový kurz* [online]. 2012 [cit. 2013-06-11]. Dostupné z: http://wp7.garvis.cz/tema-zaklady.htm

[11]MSDN: Microsoft. In: *MSDN: Microsoft* [online]. Microsoft, 2011, 21.10.2011 [cit. 2013-06-10]. Dostupné z: http://msdn.microsoft.com/en-us/library/gg680259(v=pandp.11).aspx

[12]MSDN: Microsoft. In: *MSDN: Microsoft* [online]. Microsoft, 2011, 21.10.2011 [cit. 2013-06-10]. Dostupné z: http://msdn.microsoft.com/en-us/library/gg680267(v=pandp.11).aspx

14 Seznam příloh

[1] Zdrojové kódy a soubory aplikace pro Katedru informatiky Pedagogické fakulty Jihočeské univerzity na přiloženém CD.